





PersonaSync: Refined Project Plan

Goal: A browser extension that adds a user-trainable, culturally-contextual visual persona layer to LLMs like ChatGPT, with reinforcement learning powered by Brain.js.

♦ Phase 0: Foundation (Weeks 1-2)

Objective: Prepare dev environment, define core schemas, and choose libraries.






Milestones:

-  **Dev Setup**
 - Chrome extension scaffold with Manifest V3
 - React + Three.js (or fallback Canvas2D)
 -  **Library Decisions**
 - Use Brain.js for client-side learning
 - Visual Layer: Canvas2D first, then optional GLTF + Three.js
 -  **Persona File Format**
 - .persona file: {visual_assets, brain_config, training_parameters, metadata}
 -  **Basic Certification System**
 - Centralized server for domain whitelisting (Node.js/Express)
-

♦ Phase 1: MVP – Reactive Visual Layer (Weeks 3-6)

Objective: Core loop working – visualize and train a persona.

Core Features:

-  **Brain.js Integration (v1)**
 - Lightweight Feedforward NN with manual features (e.g. sentiment, tone keywords)
-  **Canvas2D Visual Layer**
 - Reacts to Brain.js outputs (color, shape movement, particle speed)
-  **Prompt Feature Extractor**
 - Convert user/LLM text to numerical vectors (e.g. sentiment, keyword presence)
-  **User Feedback UI**
 - Popup sliders (e.g., “reaction intensity”, “formality preference”)
-  **Local Save/Load**
 - Save persona state (visual + brain config) to chrome.storage.local






Removed (for now):

- 3D GLTF rendering
 - LLM modification hooks (deferred to Phase 3)
-

♦ Phase 2: Thought Bubbles + Training (Weeks 7-10)

Objective: Add proactive suggestions via thought bubbles, deepen learning logic.




Core Features:

-  **Thought Bubble API**
 - `showThoughtBubble(text, type, actions, durationMs)`
 - e.g., "Consider softening your tone?" with Accept/Dismiss
 -  **Intervention Network**
 - Second Brain.js model for analyzing user's *prompt* (vs response)
 -  **Reinforcement Loop**
 - User feedback updates the model in real-time (e.g., click 👍 = positive reinforcement)
 -  **Experience Buffer**
 - Temporarily stores training data for batch updates
 -  **Persona Export/Import**
 - JSON + zip with assets, Brain.js weights, and training metadata
-

♦ Phase 3: Community Tools (Weeks 11-14)

Objective: Sharing, marketplace link, and plug-in support.

Core Features:

-  **AGORA Placeholder**
 - Link to future marketplace (UI only for now)
-  **Upload & Download Personas**
 - Users share .persona files locally
-  **Plugin API v1**
 - Load `persona_plugin.js` with `init()`, `update()`, `render()` hooks
 - Enables visual and behavior plug-ins

Optional (if time permits):

- Switchable visual layer modes (Canvas2D ↔ GLTF)
- Centralized server for shared weight diffs (privacy-preserving)

♦ Phase 4: Ecosystem Scaling (Post-MVP / Future Work)

Objective: Grow the product into a platform.

Stretch Goals:

- 🌐 **Federated Learning (v0.1)**
 - Share encrypted diffs P2P or via opt-in server
- 📁 **Certified Personas**
 - Persona signing for creator authenticity
- 🧩 **Marketplace AGORA**
 - Web app to browse, rate, and purchase personas
- 🎨 **Creator Tools**
 - GUI to visually adjust and preview persona reactions
- 🔗 **Decentralized Verification**
 - Explore DIDs or NFTs for ownership + authenticity

♦ Key Simplifications/Changes from Original Plan:

Feature	Kept	Modified	Removed/Deferred
Brain.js Training	✓	Added experience buffer	—
3D Visuals	✓	Phase 2+, optional	MVP uses Canvas2D
Thought Bubble UI	✓	Clarified API	—
LLM Prompt Alteration	—	Deferred to Phase 3	⌚
Federated Learning	✓	Future-proofed API	Deferred
Marketplace AGORA	✓	Placeholder MVP only	Monetization later

🕒 Time Budget (2–4 hrs/day solo dev)

Phase	Duration	Tasks
0	2 weeks	Setup, research, file structure
1	3 weeks	MVP extension with Canvas2D, Brain.js
2	3 weeks	Thought bubble logic, dual NNs
3	2 weeks	Sharing tools, plugin API
4	Ongoing	Monetization, federation, 3D visuals

🧩 Core Components & Architecture (PersonaSync)

1. Extension Shell

📌 Purpose:

Serves as the container for PersonaSync, injects the visual and behavioral layers into LLM interfaces.

📐 Architecture:

```
scss
CopyEdit
manifest.json (Manifest V3)
├── background.js (Service Worker)
├── contentScript.js (Injected into LLM pages)
├── popup.html + popup.js (User training controls)
└── persona_plugin.js (Visual Layer API Host)
```

🔧 Responsibilities:

- Detects certified LLM domains.
 - Injects `<div id="persona-root">` container.
 - Manages user permissions and onboarding.
 - Handles communication between UI, storage, and injected scripts.
-

2. Brain.js Neural Layers

📌 Purpose:

Trains and executes user-intervened logic for:

- Visual reactivity (response interpretation).
- Thought bubble suggestions (prompt intervention).

📐 Architecture:

```
CopyEdit
brain/
├── visualBrain.js (Response → Visual)
├── interventionBrain.js (Prompt → Suggestion)
├── featureExtractor.js
└── reinforcement.js
```

🔧 Responsibilities:

- Uses `Brain.js NeuralNetwork()` or `recurrent.LSTMTimestep` (optional).
- Accepts engineered features from LLM text (see next section).

- Outputs values mapped to visual parameters or suggestion triggers.
 - Receives reinforcement signals from UI (sliders, feedback buttons).
-

3. Feature Extraction Layer

Purpose:

Converts raw user prompts and LLM responses into normalized numerical vectors for the NNs.

Architecture:

```
js
CopyEdit
{
  sentiment: 0.8,
  tone_formality: 0.3,
  length: 0.5,
  question_likelihood: 0.9,
  culture_score_japanese: 0.7,
  has_slang: 0.1
}
```

Engineered Features:

- Sentiment score (via AFINN or Sentiment.js)
- Formality score (via heuristics like contractions, honorifics)
- Text length normalization
- Presence of cultural idioms/slang
- Keyword presence (directness, softness, etc.)
- Emoji frequency / punctuation style

Responsibilities:

- Modular JS module featureExtractor.js
 - Called on each new message from LLM/user
-

4. Visual Layer API (Canvas2D or Three.js)

Purpose:

Renders the visual “persona” that reacts to LLM interaction.

Architecture:

```
scss
CopyEdit
persona_plugin.js
├─ init(container)
├─ update(neuralOutput)
└─ render()
```

└─ showThoughtBubble(text, type, actions, durationMs)

Responsibilities:

- Abstract interface; plug-in based
 - Canvas2D: Color, shape, animation speed based on neural output
 - Three.js: Maps outputs to GLTF blend shapes or bone transforms
 - Thought bubbles: HTML overlays with optional interactivity
-

5. User Feedback Layer (Popup UI)

Purpose:

Allows user to train their persona by giving positive/negative reinforcement or adjusting parameters.

Architecture:

```
yaml
CopyEdit
popup.html + popup.js
├─ Sliders: e.g., "Formality Preference", "Emotional Intensity"
├─ Buttons: 👍 / 👎
└─ Settings (Save Persona, Export)
```

Responsibilities:

- Sliders send target outputs for current input features
 - Buttons reinforce current prediction as good/bad
 - Calls `reinforce(originalInput, userAction)` → modifies NN weights
-

6. Persistence Layer

Purpose:

Saves user's persona (visual, neural, settings) between sessions.

Architecture:

```
pgsql
CopyEdit
chrome.storage.local
{
  personas: {
    current: {
      visualAssets,
      neuralNetJSON,
      trainingParams,
      metadata
    }
  }
}
```

Responsibilities:

- Save/load brain state via `net.toJSON()` / `net.fromJSON()`
 - Optionally compress weights using LZ-String
 - Import/export `.persona` file as downloadable JSON bundle
-

7. Thought Bubble Engine

Purpose:

Proactively suggests prompt rewrites or tone shifts based on user input before sending to LLM.

Architecture:

```
js
CopyEdit
if (interventionBrain.run(features).intervene > threshold) {
  showThoughtBubble("Consider softening your tone", "suggestion", [...])
}
```

Responsibilities:

- Triggered on user input
 - Uses Brain.js network trained on prior interventions
 - Bubbles include:
 - Explanation text
 - Suggested rephrase
 - Action buttons ("Use", "Dismiss")
-

8. Certification System (Phase 1: Centralized)

Purpose:

Validates whether the extension is active only on certified domains (e.g., ChatGPT, Claude).

Architecture:

```
pgsql
CopyEdit
certification-server.js (Node.js)
├── /verify-domain (GET)
└── /sign-persona (POST)
```

Responsibilities:

- Responds to extension with `{ verified: true }` if domain whitelisted
 - Later can be extended with JWT verification or DID tokens
-

9. Plugin API (Optional for Phase 2+)

Purpose:

Allows community to build custom visual or behavioral persona layers.

Architecture:

```
js
CopyEdit
persona_plugin.js
export function init(container) {}
export function update(brainOutput) {}
export function render() {}
export function teardown() {}
```

Responsibilities:

- Standardized lifecycle hooks
 - Enables third-party visuals to respond to LLM and user interaction
 - Optional hooks for thoughtBubble triggers
-

Data Flow Overview

```
mermaid
CopyEdit
graph TD
  U[User Input] --> F1[Feature Extraction (Prompt)]
  F1 --> IB[Intervention Brain.js]
  IB -->|Score > Threshold| TB[Thought Bubble]

  LLM --> F2[Feature Extraction (LLM Response)]
  F2 --> VB[Visual Brain.js]
  VB --> V[Visual Layer]

  U -->|Reinforce| R[Reinforcement Handler]
  R --> VB
  R --> IB
```